Contents

Getting Started

8 minute read ✔ page test

This guide lets you quickly evaluate Istio. If you are already familiar with Istio or interested in installing other configuration profiles or advanced deployment models, refer to our which Istio installation method should I use? FAQ page.

These steps require you to have a cluster running a supported version of Kubernetes (1.19, 1.20, 1.21, 1.22). You can use any

supported platform, for example Minikube or others specified by the platform-specific setup instructions.

Follow these steps to get started with Istio:

1. Download and install Istio
2. Deploy the sample application
3. Open the application to outside traffic
4. View the dashboard

Download Istio

1. Go to the Istio release page to download the installation file for your OS, or download and extract the latest release automatically (Linux or macOS):

```
$ curl -L https://istio.io/downloadIstio | sh -
```

The command above downloads the latest release (numerically) of Istio. You can pass variables on the command line to download a specific version or to override the processor architecture. For example, to download Istio 1.6.8 for the x86_64 architecture, run:

```
$ curl -L https://istio.io/downloadIstio | ISTIO_VERSION=1.6
.8 TARGET_ARCH=x86_64 sh -
```

2. Move to the Istio package directory. For example, if the package is `istio-1.11.3`:

```
$ cd istio-1.11.3
```

   The installation directory contains:

   - Sample applications in `samples/`
   - The `istioctl` client binary in the `bin/` directory.

3. Add the `istioctl` client to your path (Linux or macOS):

```
$ export PATH=$PWD/bin:$PATH
```

Install Istio

1. For this installation, we use the `demo` configuration profile. It's selected to have a good set of defaults for testing, but there are other profiles for production or performance testing.

If your platform has a vendor-specific configuration profile, e.g., Openshift, use it in the following command, instead of the `demo` profile. Refer to your platform instructions for details.

```
$ istioctl install --set profile=demo -y
✔ Istio core installed
✔ Istiod installed
✔ Egress gateways installed
✔ Ingress gateways installed
✔ Installation complete
```

2. Add a namespace label to instruct Istio to automatically inject Envoy sidecar proxies when you deploy your application later:

```
$ kubectl label namespace default istio-injection=enabled
namespace/default labeled
```

Deploy the sample application

1. Deploy the `Bookinfo` sample application:

```
$ kubectl apply -f @samples/bookinfo/platform/kube/bookinfo.yaml@
service/details created
serviceaccount/bookinfo-details created
deployment.apps/details-v1 created
service/ratings created
serviceaccount/bookinfo-ratings created
deployment.apps/ratings-v1 created
service/reviews created
serviceaccount/bookinfo-reviews created
deployment.apps/reviews-v1 created
deployment.apps/reviews-v2 created
deployment.apps/reviews-v3 created
service/productpage created
serviceaccount/bookinfo-productpage created
deployment.apps/productpage-v1 created
```

2. The application will start. As each pod becomes ready, the Istio sidecar will be deployed along with it.

```
$ kubectl get services
NAME          TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)    AGE
details       ClusterIP   10.0.0.212      <none>        9080/TCP   29s
kubernetes    ClusterIP   10.0.0.1        <none>        443/TCP    25m
productpage   ClusterIP   10.0.0.57       <none>        9080/TCP   28s
ratings       ClusterIP   10.0.0.33       <none>        9080/TCP   29s
reviews       ClusterIP   10.0.0.28       <none>        9080/TCP   29s
```

and

```
$ kubectl get pods
NAME                              READY   STATUS    RESTARTS   AGE
details-v1-558b8b4b76-2llld       2/2     Running   0          2m41s
productpage-v1-6987489c74-lpkgl   2/2     Running   0          2m40s
ratings-v1-7dc98c7588-vzftc       2/2     Running   0          2m41s
reviews-v1-7f99cc4496-gdxfn       2/2     Running   0          2m41s
reviews-v2-7d79d5bd5d-8zzqd       2/2     Running   0          2m41s
reviews-v3-7dbcdcbc56-m8dph       2/2     Running   0          2m41s
```

Re-run the previous command and wait until all pods report READY `2/2` and STATUS `Running` before you go to the next step. This might take a few minutes depending on your platform.

3. Verify everything is working correctly up to this point. Run this command to see if the app is running inside the cluster and serving HTML pages by checking for the page title in the response:

```
$ kubectl exec "$(kubectl get pod -l app=ratings -o jsonpath='{.items[
0].metadata.name}')" -c ratings -- curl -sS productpage:9080/productpa
ge | grep -o "<title>.*</title>"
<title>Simple Bookstore App</title>
```

Open the application to outside traffic

The Bookinfo application is deployed but not accessible from the outside. To make it accessible, you need to create an Istio Ingress Gateway, which maps a path to a route at the edge of your mesh.

1. Associate this application with the Istio gateway:

```
$ kubectl apply -f @samples/bookinfo/networking/bookinfo-gateway.yaml@
gateway.networking.istio.io/bookinfo-gateway created
virtualservice.networking.istio.io/bookinfo created
```

2. Ensure that there are no issues with the configuration:

```
$ istioctl analyze
✔ No validation issues found when analyzing namespace: default.
```

Determining the ingress IP and ports

Follow these instructions to set the `INGRESS_HOST` and `INGRESS_PORT` variables for accessing the gateway. Use the tabs to choose the instructions for your chosen platform:

Minikube | Other platforms

Set the ingress ports:

```
$ export INGRESS_PORT=$(kubectl -n istio-system get service istio-in
gressgateway -o jsonpath='{.spec.ports[?(@.name=="http2")].nodePort}
')
$ export SECURE_INGRESS_PORT=$(kubectl -n istio-system get service i
stio-ingressgateway -o jsonpath='{.spec.ports[?(@.name=="https")].no
dePort}')
```

Ensure a port was successfully assigned to each environment variable:

```
$ echo "$INGRESS_PORT"
32194
```

```
$ echo "$SECURE_INGRESS_PORT"
31632
```

Set the ingress IP:

```
$ export INGRESS_HOST=$(minikube ip)
```

Ensure an IP address was successfully assigned to the environment variable:

```
$ echo "$INGRESS_HOST"
192.168.4.102
```

Run this command in a new terminal window to start a Minikube tunnel that sends traffic to your Istio Ingress Gateway:

```
$ minikube tunnel
```

1. Set `GATEWAY_URL`:

```
$ export GATEWAY_URL=$INGRESS_HOST:$INGRESS_PORT
```

2. Ensure an IP address and port were successfully assigned to the environment variable:

```
$ echo "$GATEWAY_URL"
192.168.99.100:32194
```

Verify external access

Confirm that the Bookinfo application is accessible from outside by viewing the Bookinfo product page using a browser.

1. Run the following command to retrieve the external

address of the Bookinfo application.

```
$ echo "http://$GATEWAY_URL/productpage"
```

2. Paste the output from the previous command into your web browser and confirm that the Bookinfo product page is displayed.

View the dashboard

Istio integrates with several different telemetry applications. These can help you gain an understanding of the structure of your service mesh, display the topology of the mesh, and

analyze the health of your mesh.

Use the following instructions to deploy the Kiali dashboard, along with Prometheus, Grafana, and Jaeger.

1. Install Kiali and the other addons and wait for them to be deployed.

```
$ kubectl apply -f samples/addons
$ kubectl rollout status deployment/kiali -n istio-system
Waiting for deployment "kiali" rollout to finish: 0 of 1 updated repli
cas are available...
deployment "kiali" successfully rolled out
```

If there are errors trying to install the addons, try running the command again. There may be some

timing issues which will be resolved when the command is run again.

2. Access the Kiali dashboard.

```
$ istioctl dashboard kiali
```

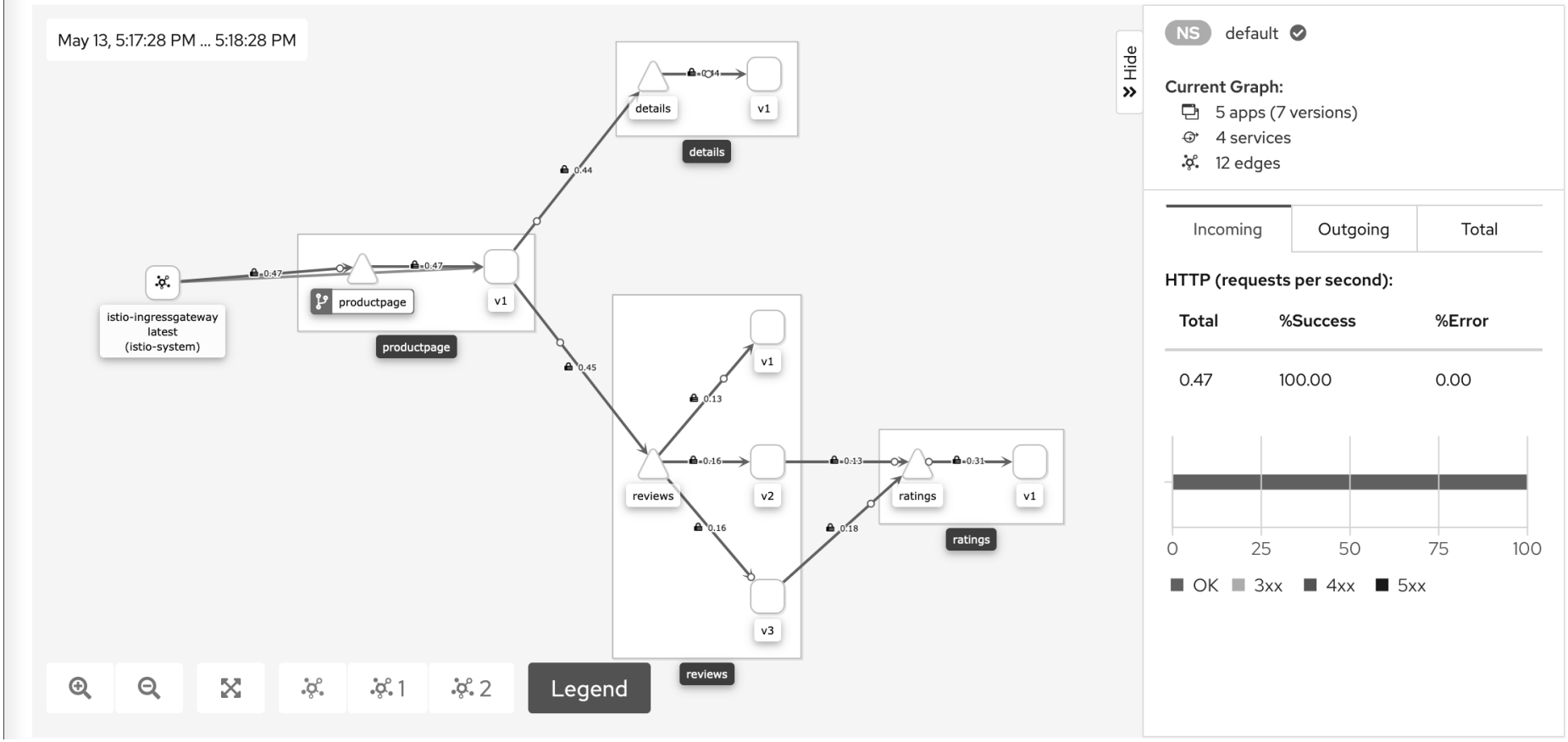
3. In the left navigation menu, select *Graph* and in the *Namespace* drop down, select *default*.

To see trace data, you must send requests to your service. The number of requests depends on Istio's sampling rate. You set this rate when you install Istio. The default sampling rate is 1%. You

need to send at least 100 requests before the first trace is visible. To send a 100 requests to the `productpage` service, use the following command:

```
$ for i in $(seq 1 100); do curl -s -o /dev/null "http://$GA
TEWAY_URL/productpage"; done
```

The Kiali dashboard shows an overview of your mesh with the relationships between the services in the `Bookinfo` sample application. It also provides filters to visualize the traffic flow.



Kiali Dashboard

Next steps

Congratulations on completing the evaluation installation!

These tasks are a great place for beginners to further evaluate Istio's features using this `demo` installation:

- Request routing
- Fault injection
- Traffic shifting
- Querying metrics
- Visualizing metrics
- Accessing external services
- Visualizing your mesh

Before you customize Istio for production use, see these

resources:

- Deployment models
- Deployment best practices
- Pod requirements
- General installation instructions

Join the Istio community

We welcome you to ask questions and give us feedback by joining the Istio community.

Uninstall

To delete the `Bookinfo` sample application and its configuration, see `Bookinfo cleanup`.

The Istio uninstall deletes the RBAC permissions and all resources hierarchically under the `istio-system` namespace. It is safe to ignore errors for non-existent resources because they may have been deleted hierarchically.

```
$ kubectl delete -f @samples/addons@
$ istioctl manifest generate --set profile=demo | kubectl delete --ignore-n
ot-found=true -f -
```

The `istio-system` namespace is not removed by default. If no

longer needed, use the following command to remove it:

```
$ kubectl delete namespace istio-system
```

The label to instruct Istio to automatically inject Envoy sidecar proxies is not removed by default. If no longer needed, use the following command to remove it:

```
$ kubectl label namespace default istio-injection-
```